# REPORT SAE1.01

BARBIERI Hugo, TSIGRIS Nicolas

Groupe D1

*18 janvier 2024*

Université Grenoble Alpes
Département INFO

## Table of contents

## Figures table

# 1. Introduction

The aim of this project was to develop a news classification system based on predefined categories. The system uses a notation system based on either manually written or automatically generated lexicons and was implemented in Java. The main objectives were to classify dispatches into predefined categories and to evaluate the system's performance.

A small example with this dispatch:

"Look at this musician who has based his tour on his dancing during his shows. "

In this sentence, you can see that some words are the same as those in the Lexicon.

- Dance : 1
- Show : 2
- Musician : 3
- Tour : 2
- Museum : 3
- Photography : 2

So, we can count the points this sentence gets for this cultural lexicon.

# 2. Features

## 2.1. Implementation

Several points were put in place :

- Reading and analysis of dispatches from a file.
- Manual creation of lexicons for the Environment-Science, Culture, Economy, Politics and Sports categories.
- Lexicon-based scoring for news categorization.
- Automatic generation of lexicons from news content.
- Classification of news articles using manual or automatic lexicons.
- Performance evaluation through runtime analysis.

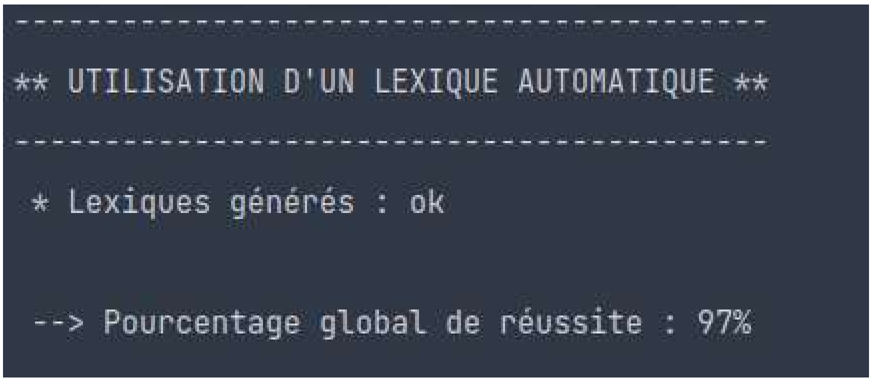
### 2.2. Not implemented

However, the use of RSS feeds could not be implemented properly. Unfortunately, lexicon generation isn't perfect when you try to use RSS feeds, giving completely inconsistent results. Especially when using the optimized version of the program.

## 3. Results

The classification system demonstrated effective categorization of news articles using both manual and automatic lexicons. The results were analyzed in terms of classification accuracy and the percentage of correct categorizations for each predefined category.

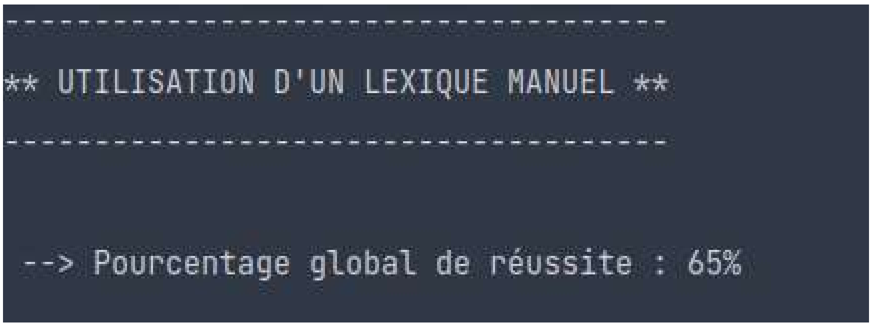
Manual lexicons provided low classification accuracy, as they are not precise enough. However, the automatic lexicons, generated on the basis of the content of the news articles, gave promising results, indicating the program's ability to define fair weights for each important word in the dispatches of a category.



```
------------------------------------------
** UTILISATION D'UN LEXIQUE AUTOMATIQUE **
------------------------------------------
 * Lexiques générés : ok

 --> Pourcentage global de réussite : 97%
```

*Figure 1 : Success rate with automatic lexicon*



```
-------------------------------------
** UTILISATION D'UN LEXIQUE MANUEL **
-------------------------------------


 --> Pourcentage global de réussite : 65%
```

*Figure 2 : Success rate with manual lexicon*

# 4. Execution time

In the original code, the classificationDepeches method is not at all optimized, nor is calculScores, due to nested loops or the over-creation of PaireChaineEntier, which take a long time to execute. For example, the original program (see Classification.java) takes an average of 2225ms to execute. To remedy this, we have used arrays; the aim is to perform as few calculations as possible, reusing results already calculated. The result is an execution time of 1487ms (see Classification_Optimise.java). This is obviously better than the previous program.

However, it would be possible to improve the program even further by using hashMaps.

# 5. Complexity analysis

The dispatch score method has a complexity of O(n), since it compares words by word. The only thing that counts in this method is the for; we count its number of occurrences. The method isn't very complex, even if the complexity could be lower, it's still acceptable.

The calculScore method has 3 nested loops, which means that complexity will depend on the size of the ArrayList present in the for loop condition. We obtain a complexity of O(n) for the first loop, O(m) for the second and O(o) for the last. This gives a total complexity of O(n*m*o), which can quickly become very high, so this method isn't very well optimized.

As you can see, the size of the ArrayList is the factor that determines the complexity of each method. Complexity can easily become very high as ArrayList size increases considerably.

# 6. Conclusion

The project successfully implemented a text classification system using manual and automatic lexicons. The results indicate the system's ability to adapt to varied content. However, there is still room for improvement in terms of efficiency and accuracy.

To improve the system, future enhancements could include :

- Integrating machine learning techniques for lexicon generation.
- Refining lexicon weights according to word importance.

- Creating a user interface for greater ease of use.

In conclusion, the system lays a solid foundation for a text classification application, and further improvements can lead to a more robust and accurate solution.